Previsão de Estágio da Infecção por COVID-19

Pedro Gigeck Freire - 10737136

MAC0425 - Inteligência Artificial

IME - USP

19/07/2020

RESUMO

Com uma extensa base de dados fornecida por alguns hospitais de São Paulo [1], montamos uma rede neural para prever o estágio de infecção por COVID-19 em algum paciente. Os estágios considerados foram: o início da infecção, detectado por um exame do tipo PCR; o começo da reação do sistema imunológico, detectado por anticorpos do tipo IgM; e o terceiro estágio, quando o paciente já está curado e produziu anticorpos do tipo IgG.

Após a limpeza e um processamento intenso dos dados, montamos a entrada da rede neural como um conjunto de dados do paciente, obtidos em uma data específica. Dentre esses dados, estão resultados de diversos exames. Então treinamos a rede neural para encontrar uma relação entre resultados dos demais exames e os estágios da infecção por COVID-19.

Os resultados obtidos foram promissores, mas inconclusivos. Após o processamento dos dados, obtemos poucas instâncias para treinamento, comparado com a quantidade original. Assim, obtemos cerca de 80% de precisão, que pode indicar a existência de uma relação entre a entrada e saída, mas também pode ser resultado de enviesamento dos dados e/ou do treinamento.

INTRODUÇÃO

A COVID-19 transformou o mundo em 2020 e afetou todos os setores da sociedade, incluindo o mundo acadêmico computacional. Com isso, nos mobilizamos para entender mais sobre a doença e seus possíveis reflexos no metabolismo humano, estimulados pelas generosas bases de dados cedidas pelos hospitais de São Paulo.

Além disso, fomos motivados pela oportunidade ímpar de aplicar os conceitos teóricos de redes neurais e aprendizado supervisionado em um tema atual e relevante. Assim, encaramos nosso empenho como um grande exercício de Ciência da Computação, não só de inteligência artificial, mas também de ciência de dados, engenharia de software e desempenho computacional.

O principal objetivo é encontrar relações entre resultados de diversos exames e a infecção por COVID-19 em três diferentes estágios. Então, dados resultados de uma bateria de exames de um paciente, busca-se prever qual estágio da infecção por COVID-19 nesse paciente. (Sendo os estágios: inicial (detecção do vírus), intermediário (anticorpos IgM) e curado (anticorpos IgG)).

A seguir, descreve-se as metodologias adotadas, com destaque para o massivo processamento dos dados. Depois, são detalhados os resultados obtidos, apresentando as planilhas e gráficos gerados. Por fim, com os resultados em mãos, faz-se uma discussão sobre a viabilidade, a precisão, as virtudes e dificuldades enfrentadas no desenvolvimento deste trabalho.

METODOLOGIA

PROCESSAMENTO DOS DADOS

Os dados brutos foram extraídos da base publicada por 3 hospitais de São Paulo em parceria com a FAPESP [1]. Nesse repositório, para cada hospital, há uma lista de pacientes, e uma extensa tabela que associa, em cada linha, um paciente com um resultado de um exame realizado em uma determinada data.

O objetivo é transformar esse padrão em uma tabela que associa, em cada linha, um paciente com vários exames realizados em uma determinada data. Os desafios são escolher quais exames serão tabelados, selecionar os dados de acordo com nossa escolha e decidir como lidar com dados faltantes.

3 BASES DE DADOS

A ideia inicial era incorporar as 3 bases em um massivo banco de dados. Porém, cada hospital possui um padrão muito diferente dos outros (codificação do arquivo, nomes dos exames, nome dos exames, formato dos dados, valores de referências, unidades de medida, entre outros). Após algum esforço tentando unificá-las, foi obtida uma tabela com mais de 2000 tipos de exames diferentes, que precisariam ser normalizados manualmente. Portanto, optou-se por usar apenas uma das bases.

Para escolher qual base seria usada, foi feito uma breve análise de cada uma com o seguinte resultado:

	Pacientes	Exames	Tipos de Exames diferentes
Einstein	4562	1800000	127
Fleury	130000	2500000	978
Sírio Libanês	2732	370000	652

Consideramos como exame um analito, isto é, a substância que o exame quer detectar ou medir, independente do método de coleta.

Assim, viu-se que o hospital Sírio Libanês possuía quase 10 vezes menos dados que as outras, já a base do Fleury tinha muitos tipos de exames diferentes, exigindo um trabalho maior de escolha e padronização dos exames. Assim, foi escolhida a base de dados do hospital Albert Einstein, por possuir menos tipos de exames diferentes, implicando em uma tabela mais densa (com menos dados nulos).

LIMPEZA DOS DADOS

Toda o processamento dos dados foi feito no script python em `./PRE/process_data.py`. Usando as bibliotecas `pandas` e `numpy`.

Os dados originais estão em `./PRE/dados_raw/`, alguns arquivos .csv criados durante o processamento estão em `./PRE/dados_inter/` e os dados processados, que serão usados na rede estão em `./PRE/dados_finais/`

O primeiro passo foi descartar as colunas que não seriam usadas. Do arquivo de pacientes, extraímos apenas o `id_paciente`, `sexo ` e `data de nascimento`. Da tabela de exames, extraímos `id_paciente`, `data da coleta`, `analito` e `resultado`. Os valores de referência não foram considerados, pois eles podem variar com a idade e sexo do paciente, além disso, espera-

se que a rede neural estabeleça alguma forma de valoração dos resultados (os pesos).

O próximo passo foi eliminar dados duplicados, tanto dos pacientes, quanto dos exames. Esse processo eliminou cerca 10000 linhas.

PACIENTES

Para a base de pacientes, foram feitos os processamentos:

- Mapeamento da coluna de `sexo` de F e M para 0 e 1;
- Pacientes com `data de nascimento` 'AAAA' transformada em1930. Isso foi feito com base no dicionário fornecido no repositório, que indica que o campo 'AAAA' é para pacientes que nasceram em 1930 ou antes.

EXAMES FEATURES

Para a base de exames, primeiramente, descartamos os exames que foram feitos poucas vezes. Para isso, conta-se quantas vezes cada exame aparece na base, essa contagem foi salva como um arquivo intermediário `./PRE/dados_inter/eintein_e_count.csv`.

Nessa contagem, observa-se que os exames de IgM e IgG para COVID-19 aparecem poucas vezes (pouco mais de 2000), mas como sabemos que eles têm que estar presentes como saída da rede neural, decidimos eliminar todos os exames menos frequentes que os de IgM e IgG. E considerar como features todos os exames que apareces mais vezes que os de IgM e IgG.

Como já existe o resultado booleano dos anticorpos, descartamos os resultados numéricos e consideramos os testes rápidos com o mesmo valor que os testes tradicionais.

Esse processamento de descartar todos os exames não-features está na função `process_analitos`.

RESULTADOS NUMÉRICOS

Com os exames filtrados, devemos transformar os resultados de textos para números. Fazemos isso na função `process_resultados`, essa transformação transforma todos os tipos de resultados boolenaos em 0 e 1 (Detectado, Reagente, Positivo, etc). E limpa algumas unidades como '%' ou 'litros' presentes nos resultados.

Com isso, eliminou-se todos os resultados não numéricos (cerca de 1%) que correspondiam majoritariamente a valores nulos.

AGRUPANDO EXAMES POR PACIENTE E DATA

Com os exames filtrados, faz-se o join dos pacientes com exames, agregando a data de nascimento e o sexo ao DataFrame.

Depois, o próximo passo é formatar os dados para a rede: agrupando os exames por paciente e data de coleta.

Isso é feito na função `group_exames`, lendo cada par (paciente x data) e montando uma linha do DataFrame final, essa é a etapa que consome mais tempo.

Na primeira vez que a tabela foi montada, obteu-se uma matriz extraordinariamente esparsa. Uma quantidade muito grande de pacientes fez uma quantidade muito pequena de exames. A tabela tinha pouco mais de 95000 linhas e 90 colunas (os exames features), dessas, 35000 não possuíam nenhum dos 3 exames de COVID-19. E apenas 4837 possuíam os 3 resultados. E a maioria dos pacientes tinha realizado menos que 5 exames no total.

Assim, a maneira escolhida para lidar com os dados esparsos foi montar **4 DataFrames** que usaremos em **4 redes neurais**. Uma das redes neurais considerará o enxuto conjunto de pacientes que realizou ou 3 exames, as outras 3 considerará os pacientes que realizaram apenas um dos exames, e tentará prever o resultado apenas daquele exame.

Então a função monta os data_frames apenas com pacientes que tenham realizados pelo menos 15 exames (dos 90 exames-features). Essa massiva limpeza de dados no deixou com os seguintes DataFrames (expressivamente menores que o original) (estão armazenados em `./PRE/dados_inter/eintein_grouped_*.pcr`)

Exames	Pacientes no DataFrame
PCR	7454
IGM	1086
IGG	1116
Todos	632

Após filtrar os pacientes com muitos valores nulos, restou ainda uma matriz esparsa. Filtramos, então, os exames com poucos resultados. Dos 90 iniciais, eliminamos todas as colunas que não estavam preenchidas pelo menos com metade dos dados.

Por fim, obtivemos 30 colunas por DataFrame e restava apenas preencher os dados faltantes. Na função `fill_empty` montamos um dicionário que mapeia os exames a um valor padrão para preencher.

Os 4 DataFrames totalmente processados foram colocados em `./PRE/dados_final/`

Vale notar que a base inicial do hospital Albert Eintein tinha 203Mb. Por fim, os dados processados ocupam 1,23Mb, menos de 1% do original.

ARQUITETURA DA REDE NEURAL

O modelo da rede neural está em `./NN/model.py` e herda o modelo de rede do pytorch. Os parâmetros da rede foram decididos depois de breves experimentos empíricos, culminando em uma rede com 2 camadas ocultas, ambas com tamanho 10.

O tamanho da camada de input e output depende de qual DataFrame estaremos utilizando. A quantidade de features e suas ordem podem ser observadas nos arquivos .csv usados para alimentar as redes. Para o output da rede usa os 3 exames, o tamanho é 3, para os outros é 1. A função de perda (loss) foi herdada do exercício feito em sala de aula, isto é, a Binary Cross Entropy, usando a implementação do pytorch.

Os estágios de infecção do paciente são dados pelos nomes dos exames que os detectam. A resposta para o estágio inicial (PCR) está na rede como o output 'Resultado COVID-19:'. Para o estágio intermediário (IgM) o nome é 'COVID IgM Interp' e para o estágio final 'COVID IgG Interp'.

DESCRIÇÃO DOS EXPERIMENTOS

Os treinamentos de todas as 4 redes neurais foram feitas utilizando a técnica de k-folds, com k = 10. Para isso, usou-se o auxílio da classe KFold da biblioteca `sklearn`, que separa os conjuntos de entrada e saída em k partições de tamanho n/k (n = tamanho do conjunto). Assim realizamos 10 experimentos com conjuntos de validações distintos.

Todo o treinamento está encapsulado na função `train_network` do arquivo `./NN/train.py`. E o treinamento de cada uma das 10 iterações com suas respectivas validações é feito pela classe `train_fold`.

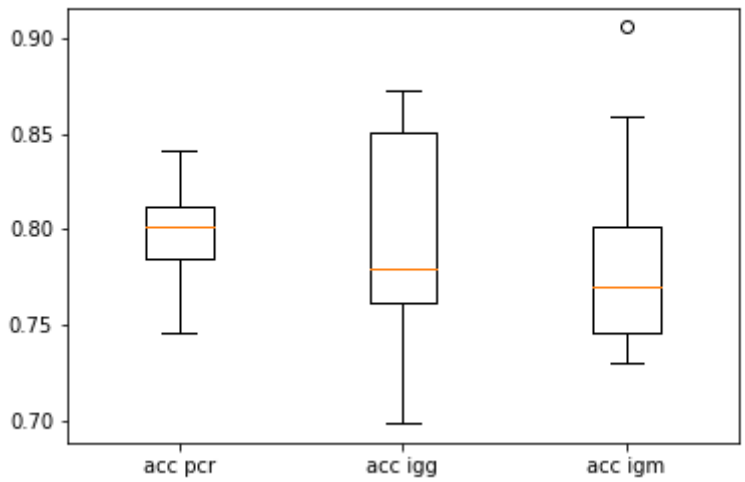
Para rodar a rede como um todo e gerar os arquivos e gráficos do experimento, basta apenas executar o arquivo `./NN/main.py`. Para gerar os dados novamente, o arquivo é `./PRE/process_data.py`.

RESULTADOS

Os resultados obtidos para cada rede neural foram salvos na seção do relatório `./REL/results/`. Lá, obtemos os resultados e valores da função de perda para cada iteração de treinamento, além de um breve relatório.

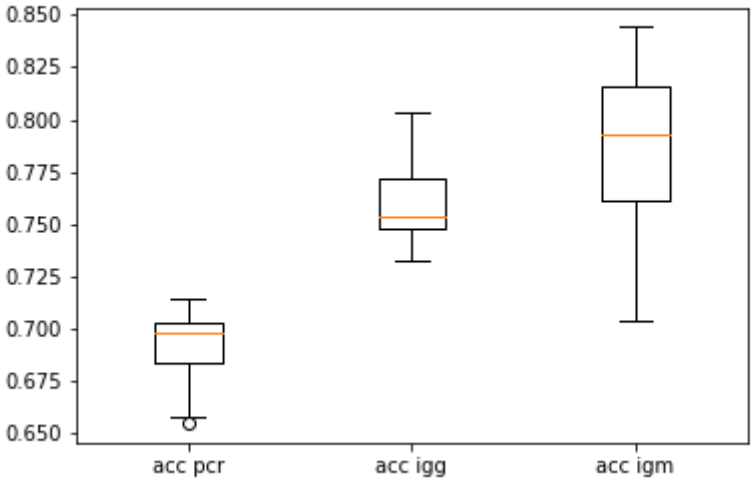
Os arquivos e relatórios são gerados em`./NN/results.py`, que também gera os seguintes gráficos

Rede neural com todos os exames



Estágio	Média	Desvio Padrão	Mínimo	Máximo
Inicial (PCR)	0.797	0.022	0.746	0.841
Curado (IgG)	0.792	0.052	0.698	0.873
Intermediário (IgM)	0.784	0.044	0.730	0.906

Redes neurais separadas para cada exame



Estágio	Média	Desvio Padrão	Mínimo	Máximo
Inicial (PCR)	0.690	0.016	0.655	0.714
Curado (IgG)	0.761	0.019	0.732	0.803
Intermediário (IgM)	0.785	0.038	0.703	0.844

DISCUSSÃO

Conforme visto na sessão do processamento dos dados, por mais expressivo que seja a base fornecida, o potencial dessa base ainda é um tanto limitado, principalmente pela pouca quantidade de exames que os pacientes tinham em comum. Assim, podíamos esperar que os resultados fossem pouco razoáveis.

Nos resultados, destacamos o resultado da rede neural específica para o exame do tipo PCR. Que é a rede que foi alimentada com mais dados e portanto mais estável. Isso reflete em um desvio padrão menor, porém com acurácia menor. Ou seja, a relação entre o exame PCR e os demais exames é fraca.

Outro fator de destaque é a alta acurácia atingida pelos valores máximos (quase 90% para a primeira rede). Indicando que pode haver alguma relação, sobretudo porque a primeira rede tem mais dados reais e menos dados preenchidos artificialmente.

O fato da precisão das nossas redes ser maior que 50% é bom. Isso quer dizer que elas são mais precisas que a mera sorte. Porém, não é razoável que um tipo teste de predição de doenças erre 30% das vezes. Assim, temos aqui um resultado promissor que pode (e deve) ser estudado mais profundamente.

Além disso, essa precisão não necessariamente reflete um caso geral, já que estamos observando pacientes de uma determinada região, de classes sociais parecidas, com uma média de idade mais elevada e que fizeram essa séries de exames por motivos indefinidos.

Todos esses fatores, junto aos resultados obtidos, indicam que existe alguma relação entre nossos dados e os valores de saída. Porém, não conhecemos quais dos nossos dados de entrada influenciaram mais para essa relação aparecer. Para extrapolar essas imprecisões, poderíamos agregar os dados dos outros hospitais, ou, principalmente, fazer um trabalho dinâmico, que considere o histórico de exames de cada paciente ao invés de um panorama de um único dia.

Em suma, os resultados mostraram que a solução proposta para o problema de previsão do estágio de infecção por COVID-19 usando uma arquitetura simples de rede neural é promissora mesmo com dados limitados. Sinaliza-se, portanto, que é viável e relevante uma solução com essa abordagem para essa e outras doenças, a fim de detectar inusitadas relações entre analitos biológicos ou contestar que essa relação não é forte o suficiente.

BIBLIOGRAFIA

[1] https://repositoriodatasharingfapesp.uspdigital.usp.br/